Periodica Polytechnica
Transportation Engineering

OnlineFirst (2017) paper 9897
https://doi.org/10.3311/PPtr.9897


RESEARCH ARTICLE

# Tuning of Look-ahead Cruise Control in HIL Vehicle Simulator

András Mihály[1*], Márk Baranyi[1], Balázs Németh[1], Péter Gáspár[1]

Received 16 August 2016; accepted 08 December 2016

***Abstract***

*The paper introduces a hardware-in-the-loop (HIL) driving simulator with the implementation of a look-ahead cruise control considering forward road information. The vehicle dynamics are simulated real-time in the high fidelity heavy duty vehicle simulation environment TruckSim, while the proposed look-ahead control algorithm also runs real-time on dSPACE MicroAutoBox II. The latter functions as a vehicle electronic control unit (ECU) and is used for rapid control prototyping (RCP), hence the proposed look-ahead driver assistance system can be tested and tuned in a real-time HIL vehicle simulator before installing dSPACE MicroAutoBox II in a real vehicle.*

***Keywords***

*HIL, look-ahead control, vehicle simulator, dSPACE*

[1] Institute for Computer Science and Control,
Hungarian Academy of Sciences,
H-1111 Budapest, Kende u. 13-17., Hungary

[*] Corresponding author, e-mail: mihaly.andras@sztaki.mta.hu

## 1 Introduction and motivation

Software-in-the-loop (SIL) and hardware-in-the-loop (HIL) simulation systems are powerful tools for developing production ready controllers. The integration of a compiled software codes in simulation systems have multiple purposes: it can increase simulation speed, verify the control algorithm in a production controller, etc. The development of SIL and HIL simulation environments are getting more and more widespread among automotive companies and researchers as well, see (Bringmann and Krämer, 2008; Vandi et al., 2014). In (Szalay et al., 2012) a HIL vehicle driving simulator has been developed based on a real vehicle. In (Gietelink et al. 2007) a HIL environment served for the development of intelligent driver assistant systems, while in (Deng et al., 2008) a laboratory environment has been developed to test and verify autonomous vehicle functions. In (Aradi et al., 2014) and (Törő et al., 2016) camera related autonomous vehicle functions have been tested in HIL environment with the intention of applying the developed algorithm on go-cart vehicle.

Present paper deals with the HIL environment of a real-time driving simulator with the aim of testing and tuning of a previously introduced look-ahead cruise controller. The parameter dependency of the operation of the look-ahead control algorithm has been already presented in earlier papers, see Németh and Gáspár (2015). The speed selection of the look-ahead cruise control has been compared to that of a human driver, as discussed in Mihály et al. (2012; 2013). The HIL environment presented in the paper enables the driver/developer to test the operation of the proposed look-ahead cruise control along with a conventional cruise control or manual driving in a real-time simulation environment.

The procedure to convert from HIL environment into a real system working on a vehicle has the following steps: First, the function of the software components are developed (here implanted into dSPACE MicroAutoBox II which is connected to TruckSim during the HIL tests). Next, communication lines valid on the real vehicle must be formed, tested. After the validation of the communication lines MicroAutoBox II can be separated from the HIL environment and inserted into the real

vehicle. This phase paves the way to prototype production. The development follows the so-called V-model because each phase fits flexibly to each other and one can return to earlier phase without any loss.

The paper is organized as follows. Section 2 shows the interface between TruckSim and dSPACE MicroAutoBox II. Section 3 introduces the simulation models running under both the desktop computer and MicroAutoBox II. Section 4 shows the operation of the real-time HIL environment. Finally, some concluding remarks are presented in Section 5.

## 2 Architecture of the TruckSim/dSPACE interface

The HIL simulation environment is shown in Fig. 1. The left side represents the vehicle simulator with the driver seat, vehicle control devices (steering wheel, brake/accelerator pedal, gearbox) and the desktop computer with the TruckSim/Simulink vehicle simulation environment, while the right side shows dSPACE MicroAutoBox II on which the proposed look-ahead cruise control algorithm is running.

TruckSim is a simulation environment with high complexity models of trucks, buses, and other heavy duty vehicles. The software package includes the Driving Simulator, which enables real-time measurements. The purpose of the proposed HIL environment is to perform real-time tests of the look-ahead controller designed for a specific heavy duty vehicle. Another aim of the presented platform, is to enable the driver/developer of the simulator to tune the parameters of the controller and observe the effect on the operation of the vehicle. TruckSim/dSPACE interface is important to ensure that the developed controller can easily be implemented on a real vehicle.

dSPACE MicroAutoBox II serves as a testing and fast function prototyping ECU. It operates real-time without user intervention and virtually may carry out any task on the vehicle. A desktop or laptop can be connected to MicroAutoBox II for software downloading, data analysis or controller calibration. Here, dSPACE MicroAutoBox II 1401 / 1511 / 1512 is used, while the main interface with the desktop computer is based on TCP/IP protocol. The CAN communication between the desktop computer and dSPACE MicroAutoBox II is provided by Vector CANboardXL.

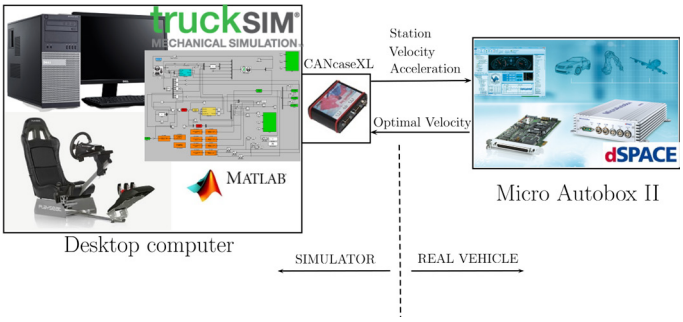



**Fig. 1** Architecture of the HIL communication interface

dSPACE ControlDesk is the ECU developing software for MicroAutoBox II. It is a universal, modular test tool kit for ECU software development. This software is used for MicroAutoBox calibration, measurement and diagnostics, and it supports synchronized data processing. During real-time HIL tests of the look-ahead control algorithm the transmitted and received CAN signals can be monitored with dSPACE ControlDesk, as detailed later with a test example in Section 4.

## 3 Simulation models

### 3.1 TruckSim/Simulink model

The MATLAB/Simulink model implemented in TruckSim real-time driving simulator environment is depicted in Fig. 2. The red blocks represent the PI speed controller, the yellow block contains the vehicle model used to generate the throttle/brake inputs for TruckSim, the green blocks define the GPS position of the vehicle, while the orange blocks stands for the connection between TruckSim and dSPACE MicroAutoBox II via CAN communication channel. The latter consists of multiple dialog blocks from MATLAB Vehicle Network Toolbox. First, the CAN Pack block transforms the measured real-time vehicle and steering wheel button signals into CAN messages with a given identifier (ID). The measured vehicle signals generated by TruckSim are the position, velocity and acceleration of the vehicle, while there are six buttons altering the three controller parameters (look-ahead distance: $L$, number of section points: $n$, performance weight: $R_1$) with a predefined scaling. Next, the CAN Transmit blocks transmit the CAN messages to MicroAutoBox II via Vector CANboardXL.

The look-ahead cruise control algorithm implemented in MicroAutoBox II then calculates the optimal velocity for the vehicle based on the received CAN signals from TruckSim/Simulink, and sends back the result to the CAN Receive block with a sampling time of $T_s = 0.01$s. Finally, the CAN Unpack block transforms the CAN message for MATLAB/Simulink and serves as the reference signal for the PI cruise controller. Note, that a CAN Configuration block is also needed in order to define the interface device (Vector CANboardXL) and bus speed.

### 3.2 dSPACE control model

The MATLAB/Simulink model of the look-ahead cruise control implemented in dSPACE MicroAutoBox II is shown in Fig. 3. There are three main parts of the model: Receiving, processing, transmitting. The receiving block obtains the data from CAN and transforms it to operable signals. The processing block gathers these signals and the algorithm calculates the optimal velocityfor the vehicle based on the look-ahead method. Finally, the last transmitting block sends information through serial and CAN interface.

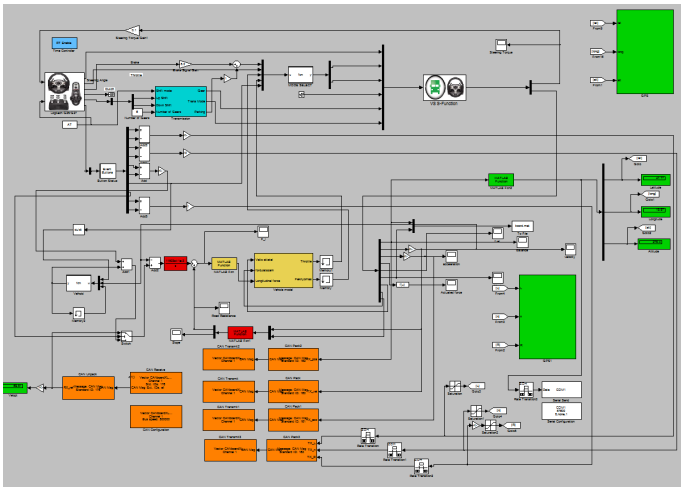


**Fig. 2** MATLAB/Simulink model

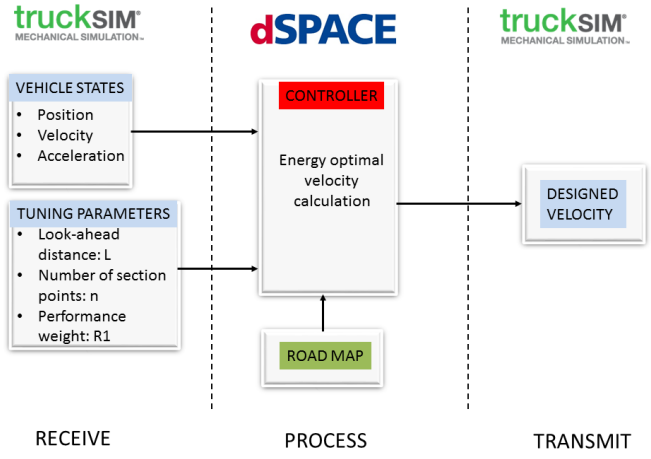



**Fig. 3** The dSPACE model

The model includes the necessary CAN and serial communication boxes, which handles Vector CANCase card. The vehicle position, velocity and acceleration data along with the algorithm parameters ($L$, $R_1$, $n$) are obtained via CAN communication. Note, that the calculated optimal velocity is also sent back to TruckSim on the same CAN communication channel. The block in the middle of Fig. 3 contains the velocity optimization, and gives an energy optimal velocity as a result. The optimal velocity calculation for given road characteristics has already been introduced in Németh and Gáspár (2011; 2013). Here, instead of the computationally cumbersome optimization methods an analytical solution is given since MicroAutobox II has limited capacity. Hence in case the algorithm cycle calculation takes too much time, the embedded operating system gives task overflow exception, which is handled with restarting the algorithm with reduced cycle size. Note, that the computational time of MicroAutobox II heavily depends on the number of section points $n$ set by the driver during the controller tuning process.

After the MATLAB/Simulink model of the look-ahead controller is set up in dSPACE ControlDesk the compiling, linking and downloading of the program code can be evaluated. Here, a sampling time of $T_s = 0.01$s. has been set. Finally, the C code of the look-ahead controller is created and can be called during the HIL simulation.

## 4 Real-time simulations

The real-time bus simulator has three driving modes, which can be activated by the use of the steering wheel buttons, as depicted in Fig. 4.

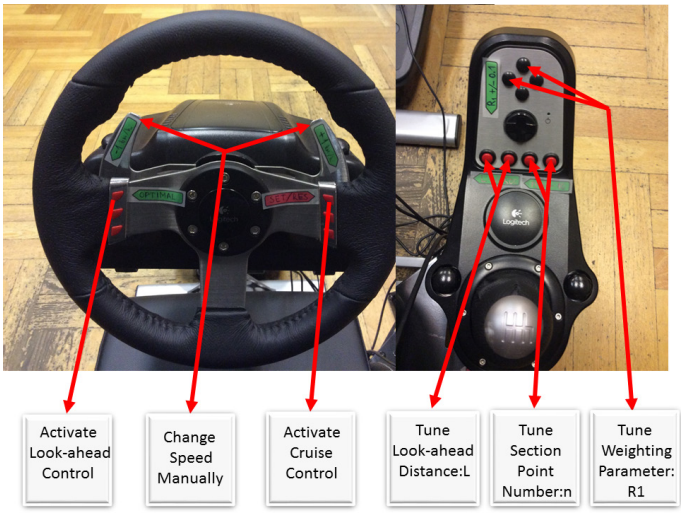
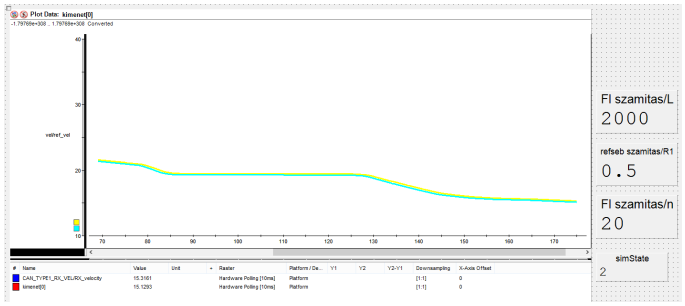



**Fig. 4** Function buttons on the Logitech steering wheel

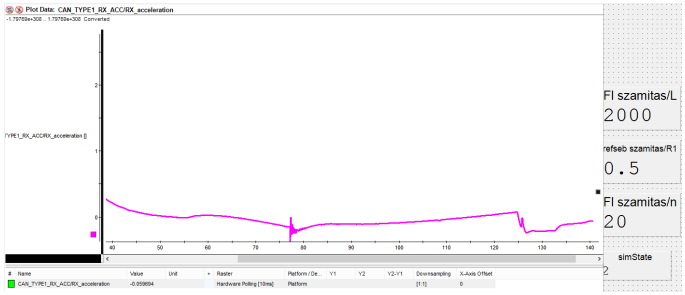
In manual driving the driver has full control over the bus, thus steering, throttle and brake pedal usage is a driver task. In this mode, the vehicle speed is also set by the driver. Pushing the SET/RESET button activates the conventional cruise control and sets the bus velocity to the actual speed. This speed can then be adjusted by pulling the left/right steering wheel paddles. This driving mode only requires steering intervention of the driver.

In case the driver pushes the OPTIMAL button on the steering wheel, the reference velocity is set by the look-ahead control algorithm running under dSPACE MicroAutoBox II. Note, that the parameters of the look-ahead controller can be adjusted real-time by the driver using gearbox console buttons shown in Fig. 4. The look-ahead distance $L \in [0,5000]$ can be adjusted with 500 m intervals, the number of division points $n \in [10,100]$ can be incremented/decremented by 10, while the optimization weight $R_1 \in [0,1]$ can also be adjusted by 0.1 steps.

The real-time CAN signals of MicroAutoBox II can be visualized in dSPACE ControlDesk. Fig. 5 shows an example of a real-time simulation section. In Fig. 5(a) the reference optimal velocity and the actual vehicle velocity is depicted, with the following tuning parameter settings: $L = 2000$m, $n = 20$m, $R_1 = 0.5$. It is well demonstrated, that the simulated bus follows the reference velocity with minimal error. In Fig. 5(b) the measured acceleration is depicted.




(a) Vehicle speed signal

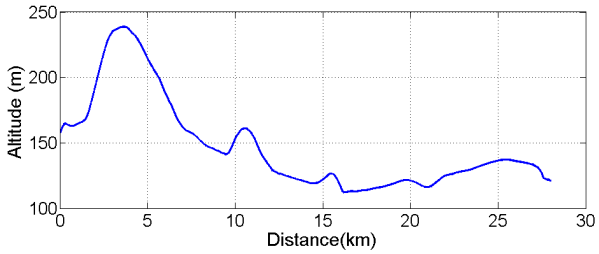



(b) Vehicle acceleration signal

**Fig. 5** Controldesk

Above real-time simulation has been performed based on the road characteristics of the Hungarian highway road section between Gyöngyös and Kápolna, see Fig. 6(a). The terrain characteristics of the road is shown in Fig. 6(b).

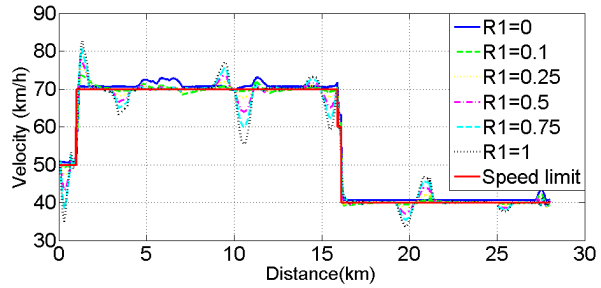

(a) Road map

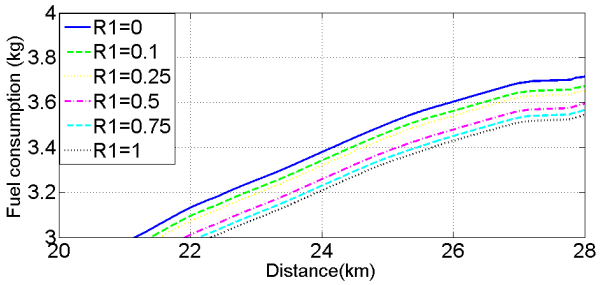



(b) Altitude

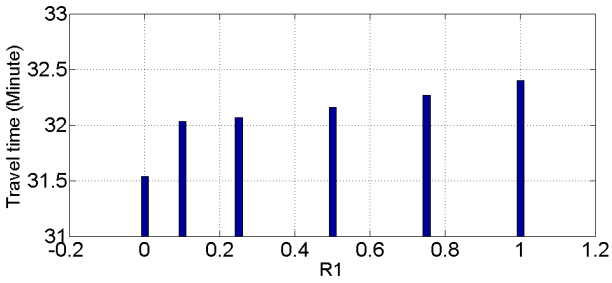
**Fig. 6** Terrain characteristics of simulation road

The effect of altering the weighting parameter $R_1$ is shown in Fig. 7. Here, the simulated bus has been driven along the selected road shown in Fig. 6 with fixed look-ahead distance ($L$ = 1000m) and number of section points ($n$ = 20) while the performance weighting parameter $R_1 \in [0,1]$ has been altered. It is well demonstrated in Fig. 7(a), that increasing $R_1$ the velocity profile of the bus differs more and more from that given by a conventional cruise controlwithout energy consideration (simulated with selecting $R_1 = 0$ meaning no forward road consideration). Also, increasing $R_1$ results in better fuel consumption (see Fig. 7(b)) while the total travel time of the journey slightly increases, as depicted in Fig. 7(c).



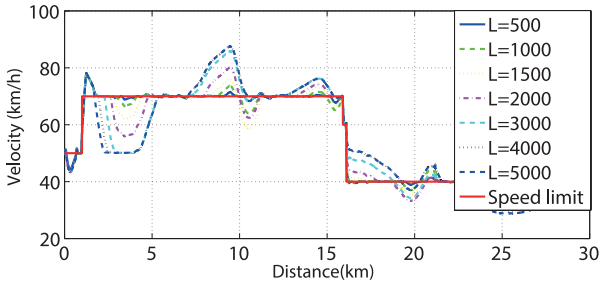

(a) Fuel consumption



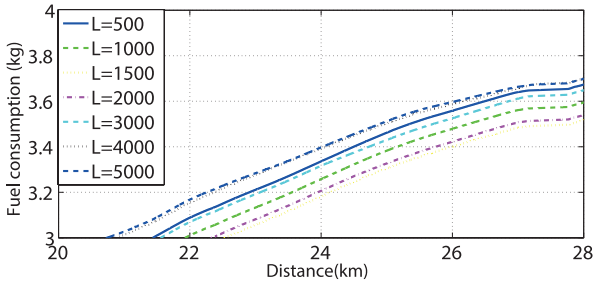

(b) Traveling time

**Fig. 7** Effect of weighting parameter $R_1$

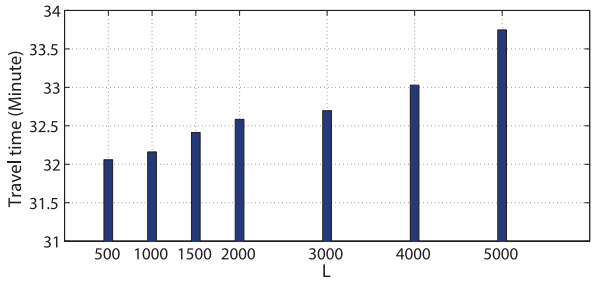
An other example of controller tuning is depicted in Figure 8. Here, the motion of the bus has been simulated with different look-ahead distance settings, but with similar segment lengths ($L$ / $n$ = 50m) and optimization weighting parameters ($R_1$ = 0.5). It is well demonstrated in Fig. 8(a), that the velocity selection of the bus greatly depends on the look-ahead distance, and generally bigger distance induces bigger deviation from the speed limit. Fig. 8(b) suggests that the overall fuel consumption is the least in case of a look-ahead distance $L$ between 1500–2000m. Although the travel time increases with the increasing look-ahead distance (see Fig. 8(c)), the difference between the two extremes is less than 2 minutes.




(a) Velocity




(b) Fuel consumption




(c) Traveling time

**Fig. 8** Effect of altering the look-ahead distance *L*

## 5 Conclusion

The paper presented a HIL driving simulator based on real-time TruckSim vehicle simulator and dSPACE MicroAutoBox II prototyping system for in-vehicle applications. The purpose of the HIL driving simulator is to develop, test and tune a look-ahead cruise controller. The real-time HIL simulation environment enables the driver/developer to adjust the parameters of look-ahead controller real-time. The simulation results suggests that calibration of the look-ahead control parameters are necessary to guaranteeoptimal performances. The presented HIL environment contributes in a fast and cheap ECU software development, which can then simply be adopted to a real vehicle for prototype testing.

## Acknowledgment

The research was supported by the National Research, Development and Innovation Fund through the project "SEPPAC: Safety and Economic Platform for Partially Automated Commercial vehicles" (VKSZ 14-1-2015-0125). This paper was partially supported by the János Bolyai Research Scholarship of the Hungarian Academy of Sciences.

## References

Aradi, S., Bécsi, T., Gáspár, P. (2014). Experimental vehicle development for testing autonomous vehicle function. In: IEEE/ASME 10th International Conference on Mechatronic and Embedded Systems and Applications (MESA). Ancona, Italy, Sep. 10-12. 2014. pp. 2014. https://doi.org/10.1109/MESA.2014.6935534

Bringmann, E., Krämer, A. (2008). Model-based testingof automotive systems. In: 1st International Conference on Software Testing, Verification, and Validation. Lillehammer, Norway, Apr. 9-11, 2008, pp. 485–493. https://doi.org/10.1109/ICST.2008.45

Deng, W., Lee, Y. H., Zhao, A. (2008). Hardware-in-the loop simulation for autonomous driving. In: IECON 2008. 34th Annual Conference of IEEE. Industrial Electronics, IECON 2008. Orlando, Florida, USA, Nov. 10-13, 2008, pp. 1742–1747. https://doi.org/10.1109/IECON.2008.4758217

Gietelink, O., Ploeg, J., De Schutter, B., Verhaegen, M. (2007). Development of advanced driver assistancesystems with vehicle hardware-in-the-loop simulations.*Vehicle System Dynamics*. 44(7), pp. 569–590. https://doi.org/10.1080/00423110600563338

Mihály, A., Németh, B., Gáspár, P. (2012). Analysis of driver behavior related to look-ahead control. *13thIFAC Symposium on Control in Transportation Systems*. 45(24), pp. 268-273. https://doi.org/10.3182/20120912-3-BG-2031.00056

Mihály, A., Németh, B., Gáspár, P. (2013). Enhancement of driver speed based on multi-criteria optimization. *Periodica Polytechnica Transportation Engineering*. 41(1), pp. 71–76. https://doi.org/10.3311/PPtr.7103

Németh, B., Gáspár, P. (2011). Road inclinations in the design of LPV-based adaptive cruise control. In: 18thIFAC World Congress. 44(1), pp. 2202-2207. https://doi.org/10.3182/20110828-6-IT-1002.00932

Németh, B., Gáspár, P. (2013). Design of vehicle cruise control using road inclinations. *International Journal of Vehicle Autonomous Systems*. 11(4), pp. 313–333. https://doi.org/10.1504/IJVAS.2013.056651

Németh, B., Gáspár, P. (2015). Model-based sensitivity analysis of the look-ahead cruise control. In: 15th IEEE International Symposium on Computational Intelligence and Informatics, (CINTI). Budapest, Hungary, Nov. 19-21, pp. 103-108. https://doi.org/10.1109/CINTI.2014.7028657

Szalay, Z., Gáspár, P., Kánya, Z., Nagy, D. (2012). Development of vehicle simulator based on a Real Car for research and education purposes. *Proceedingsof the FISITA 2012 World Automotive Congress*. 196, pp. 1301–1312. https://doi.org/ 10.1007/978-3-642-33738-3_31

Törő, O., Bécsi, T., Aradi. S. (2016). Design of lanekeeping algorithm of autonomous vehicle. *Periodica Polytechnica Transportation Engineering*. 44(1), pp. 60–68. https://doi.org/10.3311/PPtr.8177

Vandi, G., Cavina, N., Corti, E., Mancini, G., Moro, D., Ponti, F., Ravaglioli, V. (2014). Development of a software in the loop environment for automotive powertrain systems. *Energy Procedia*. 45, pp. 789-798. https://doi.org/10.1016/j.egypro.2014.01.084